CiteSeer Find: Register Allocation Pipelined loops Documents Citations

Searching for PHRASE **register allocation pipelined loops**.
Restrict to: Header Title Order by: Citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. **Only retrieving 250 documents (System busy - maximum reduced).** Retrieving documents... **Order: relevance to query.**

An Integrated Approach to Data Path Synthesis for Testability - Yang, Peng (1997) (Correct)
which is based on testability analysis at **register**-transfer level, is proposed. Contrary to other algorithm which carries out the scheduling and **allocation** tasks in an integrated fashion. In particular, clk(S i S j)end if if U involve in a feedback **loop** then CC out =CC out \Theta CC **loop** ,SC out =
ftp.ida.liu.se/pub/labs/eslab/papers/ITSW97.ps.gz

Integrated Register Allocation and Software Pipelining - Janssen, Corporaal, Karkowski (1998) (Correct)
Integrated **Register Allocation** and Software **Pipelin**ing Johan
cardit.et.tudelft.nl/MOVE/papers/Janssen98a.ps.gz

Resource Constrained Dataflow Retiming Heuristics for VLIW .. - Jacome, de Veciana, Akuran (1999) (Correct)
For Vliw Asips With Heterogenous Distributed **Register** Structures. We Discuss A Code Generation Phasing architectures. For example, performing **register allocation** and assignment in the context of the small, a transformation of the original dataflow aimed at **pipelin**ing several **loop** body iterations within the same
horizon.ece.utexas.edu/~jacome/Publications/CODES99.ps

The SPARK 2.0 system - a special purpose vector.. - Formella.. (Correct)
loops, where **loop** control is supported by a count **register**. The least significant bits of the branch is that we have neither implemented the dynamic **allocation** of storage via the new statement nor the point conditions have to fall through the **pipeline** stages. Vector operations are performed by
www-wjp.cs.uni-sb.de/~formella/hawaipos.ps.gz

Stack-Based Typed Assembly Language - Morrisett, Crary, Walker, Glew (1998) (Correct) (24 citations)
such as tail call elimination and callee-saves **registers**. This paper also formalizes the typing However, most compilers are based on stack **allocation**. This paper presents STAL, an extension of TAL compiler optimizations including inlining, **loop**-unrolling, **register allocation**, instruction
reports-archive.adm.cs.cmu.edu/anon/1998/CMU-CS-98-178.ps

Register Allocation with Instruction Scheduling: a New Approach - Pinter (1993) (Correct) (57 citations)
Register Allocation with Instruction Scheduling: a New
cardit.et.tudelft.nl/~steven/ilp/pinter93.ps.gz

LoRA: a Package for Loop Optimal Register Allocation - Eisenbeis, Lelait (1997) (Correct) (2 citations)
LoRA: a Package for **Loop** Optimal **Register Allocation** Christine Eisenbeis Sylvain Lelait
LoRA: a Package for **Loop** Optimal **Register Allocation** Christine Eisenbeis Sylvain Lelait INRIA
ftp.inria.fr/INRIA/Projects/a3/lelait/wcgep.ps.gz

RESIS: A New Methodology for Register Optimization in.. - Ferm'in Anchez (1996) (Correct) (1 citation)
RESIS: A New Methodology for **Register** Optimization in Software **Pipelin**ing Ferm'in
ftp.ac.upc.es/pub/reports/DAC/1996/UPC-DAC-1996-15.ps.Z

Compilation and Pipeline Synthesis for Reconfigurable.. - Weinhardt (1997) (Correct) (8 citations)
(length L) on N 16-bit numbers with a feedback **register** for random number generation. We chose this Compilation and **Pipeline** Synthesis for Reconfigurable Architectures
is analyzed and vectorized. Then, for suitable **loops**, hardware **pipelines** are synthesized. They execute
i44ftp.info.uni-karlsruhe.de/pub/papers/weinhardt/raw97.ps.gz

Co-Scheduling Hardware and Software Pipelines - Govindarajan, Altman, Gao (1995) (Correct) (1 citation)
to sharing of resources, e.g. Result Buses or **Register** Ports that are shared by different stages of an P. P. Tirumalai, and M. S. Schlansker. **Register allocation** for software **pipelined loops**. In Proc. of the Systems Co-Scheduling Hardware and Software **Pipelines** R. Govindarajany Erik R. Altman Guang R. Gao
ftp.capsl.udel.edu/pub/doc/acaps/memos/memo92.ps.gz

A Scheduler-Sensitive Global Register Allocator - Norris, Pollock (1993) (Correct) (12 citations)

A Scheduler-Sensitive Global **Register** Allocator Cindy Norris Lori. L. Pollock
ftp.udel.edu/pub/people/pollock/SSG.ps

An Interprocedural Parallelizing Compiler and Its Support for.. - Trung Nguyen (1995) (Correct) (5 citations)
in the trace is either a memory reference or a **register** reference. A memory reference is due to either a analysis to support task scheduling, data **allocation** and efficient memory management. In order to compiler, Panorama, to improve array privatization, **loop** parallelization and the efficiency of memory
www-users.cs.umn.edu/~gu/papers/icpc95.ps

Probabilistic Scoreboards for Superscalar Processors - Bjarne Steensgaard (Correct)
scoreboard) for use by instruction schedulers and **register** allocators for superscalar processors supporting For instruction scheduling and **register allocation** purposes, it is often convenient to have a tool for instruction level parallelism (ILP) or for **pipelined** processors where the only visible sign of the
ftp.research.microsoft.com/pub/TR/TR-95-09.ps

Techniques for Functional Test Pattern Execution - On (1997) (Correct)
The approach imposes minimal restriction on **register** sharing so that the synthesized designs will of the optimization potential when scheduling, **allocation** and binding tasks in high-level synthesis are designed for detecting design errors in **pipelined** computer implementations. An approach which
ftp.cs.ucla.edu/tech-report/97-reports/970025.ps.Z

Simple Register Spilling in a Retargetable Compiler - Fraser, Hanson (1995) (Correct) (8 citations)
VOL. 00(00)000-000 (MONTH 1995) Simple **Register** Spilling in a Retargetable Compiler CHRISTOPHER
ftp.cs.princeton.edu/pub/lcc/contrib/spills.ps.gz

A Novel Framework of Register Allocation for Software Pipelining - Ning, al. (1993) (Correct) (38 citations)
ACM (see notice below)A Novel Framework of **Register Allocation** for Software **Pipelin**ing Copyright c
ftp.capsl.udel.edu/pub/doc/acaps/papers/POPL93.ps.gz

Data Prefetching And Data Forwarding In Shared Memory.. - Poulsen, Yew (1994) (Correct) (11 citations)
considerably less complex than existing software **pipelined** prefetching algorithms, is shown to be parallel numerical application codes with **loop**-level and vector parallelism. More data, algorithms on large, numerical applications with **loop**-level and vector parallelism. Data forwarding is a
www.cs.umn.edu/Research/Agassiz/Paper/poulsen.icpp94.ps.Z

Return Value Placement And Tail Call Optimization In High.. - Bigot, Debray (1999) (Correct) (3 citations)
language implementations return output values in **registers**, while most logic programming systems return
www.cs.arizona.edu/people/debray/papers/retval_placement.ps

Baring it all to software: Raw machines - Waingold, Taylor, Srikrishna.. (1997) (Correct) (70 citations)
and data memories, an arithmetic logic unit, **registers**, configurable logic, and a programmable switch -to determine and implement the best resource **allocation** for each application. We call systems based on than today's superscalars, and support efficient **pipelined** parallelism for multimedia applications. RAW
ftp.cag.lcs.mit.edu/pub/raw/documents/computer97.ps.Z

Documents 41 to 60 Previous 20 Next 20

Try your query at: Amazon Barnes & Noble Google (RI) Google (Web) CSB DBLP

CiteSeer Find: Register Allocation Pipelined loops Documents Citations

Searching for PHRASE **register allocation pipelined loops**.
Restrict to: Header Title Order by: Citations Hubs Usage Date Try: Amazon B&N Google (RI) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. **Only retrieving 250 documents (System busy - maximum reduced).** Retrieving documents... **Order: relevance to query.**

Register Allocation over the Program Dependence Graph - Norris, Pollock (1994) (Correct) (19 citations)
Register Allocation over the Program Dependence Graph
www.eecis.udel.edu/pub/people/pollock/rap.ps

A New Fast Algorithm for Optimal Register Allocation in.. - Lelait, Gao, Eisenbeis (1998) (Correct) (1 citation)
En Automatique A New Fast Algorithm For Optimal **Register Allocation** In Modulo Scheduled **Loops** Sylvain
ftp.inria.fr/INRIA/publication/RR/RR-3337.ps.gz

The meeting graph: a new framework for loop register.. - Eisenbeis, Lelait, Marmol (1995) (Correct) (1 citation)
The meeting graph :a new framework for **loop register allocation** Christine EISENBEIS ,Sylvain LELAIT
ftp.inria.fr/INRIA/Projects/a3/lelait/RR-2756.ps.gz

Global Register Allocation Based on Graph Fusion - Guei-Yuan Lueh (1996) (Correct) (7 citations)
Global **Register Allocation** Based on Graph Fusion Guei-Yuan Lueh
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/lcpc96.ps

Linear-Time Register Allocation for a Fixed Number of.. - Bodlaender, Gustedt, Telle (1997) (Correct)
FACHBEREICH 3 MATHEMATIK Linear-Time **Register Allocation** for a Fixed Number of **Registers** by
ftp.math.tu-berlin.de/pub/Preprints/combi/Report-551-1997.ps.Z

Integer Linear Programming vs. Graph-Based Methods in.. - Kästner, Langenbach (1998) (Correct)
problem between instruction scheduling and **register allocation**. Many compilers perform these tasks
www.cs.uni-sb.de/~mlangen/papers/CGEP98.ps.gz

Resource Spackling: A Framework for Integrating Register.. - David Berson (1994) (Correct) (16 citations)
Resource Spackling: A Framework for Integrating **Register Allocation** in Local and Global Schedulers y
Spackling: A Framework for Integrating **Register Allocation** in Local and Global Schedulers y David A.
www.cs.pitt.edu/~soffa/research/Comp/pact94.ps

A Register Allocation Framework Based on Hierarchical.. - Hendren, Gao, Altman, .. (1993) (Correct) (30 citations)
Compilers, Architectures and Parallel Systems A **Register Allocation** Framework Based on Hierarchical
ftp.capsl.udel.edu/pub/doc/acaps/memos/memo33.ps.gz

Optimum Modulo Schedules for Minimum Register Requirements - Eichenberger, Davidson.. (1995) (Correct) (12 citations)
Optimum Modulo Schedules for Minimum **Register** Requirements Alexandre E. Eichenberger and
www4.ncsu.edu/~alexe/papers/ICS95.ps

Non-Consistent Dual Register Files to Reduce Register Pressure - Llosa, Valero, Ayguade (1995) (Correct) (8 citations)
Non-Consistent Dual **Register** Files to Reduce **Register** Pressure Josep Llosa,
ftp.ac.upc.es/pub/archives/hpc/Papers/josepll95j2C.ps.gz

Register Allocation Using Lazy Saves, Eager Restores, and .. - Burger, Waddell, Dybvig (1995) (Correct)
Register Allocation Using Lazy Saves, Eager Restores, and
ftp.cs.indiana.edu/pub/scheme-repository/doc/pubs/Reg-Alloc-PLDI95.ps.gz

The Performance of Scalar Replacement on the HP 715/50 - Steve Carr (Correct)
been shown that the **allocation** of array values to **registers** can significantly improve the memory
1 Introduction It has been shown that the **allocation** of array values to **registers** can significantly
scalar replacement. In each case, the software **pipelin**ing algorithm used by the HP compiler is
cs.mtu.edu/pub/carr/ScalarReplacement.ps.gz

Minimum Register Requirements for a Modulo Schedule - Alexandre Eichenberger (1994) (Correct) (4 citations)
Minimum **Register** Requirements for a Modulo Schedule Alexandre E.
integer) multiple of II. 3. **Register allocation** performs the actual **allocation** of virtual

scheduling heuristics. 1 Introduction Software **pipelining** is a technique for exploiting the
www.cs.rutgers.edu/~uli/cs671/MICRO97-Eichenberger.ps

High-Level Synthesis for Easy Testability During Data Path.. - Yang (Correct)
whose basic idea is to fold nodes (modules and **registers**) with good controllability and bad
Synthesis for Easy Testability During Data Path **Allocation** Tianruo Yang Linkoping University, Department
and observability. Additionally, since sequential **loops** are widely recognized to make a design
www.stw.nl/prorisc/workshop/proc/psz/yang4.ps.gz

Some MPEG Decoding Functions on Spert An Example for Assembly.. - Formella (1994) (Correct) (1 citation)
simple scheduling strategy and a straight forward **register allocation** algorithm. We define some lower and
strategy and a straight forward **register allocation** algorithm. We define some lower and an upper
eight units ffl the functional units are fully **pipelined** and can work in parallel on the **register** file,
ftp.icsi.berkeley.edu/pub/techreports/1994/tr-94-027.ps.gz

Evicted Variables and the Interaction of Global Register.. - Ali-Reza Adl-Tabatabai (Correct)
Evicted Variables and the Interaction of Global **Register Allocation** and Symbolic Debugging Ali-Reza
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/popl93.ps

Optimal Contiguous Expression DAG Evaluations - Keßler, Rauber (Correct)
for expression DAGs with a minimal number of **registers** is NP-complete. We present two algorithms
evaluations quite fast. 1 Introduction **Register allocation** is one of the most important problems in
from the application of compiler techniques like **loop** unrolling [6] and trace scheduling [7]Therefore,
www.informatik.uni-trier.de/~kessler/regalloc/a3fct.ps

Generalizations of the Sethi-Ullman algorithm for register.. - Appel, Supowit (1987) (Correct) (6 citations)
Generalizations of the Sethi-Ullman algorithm for **register allocation** Andrew W. Appel Kenneth J. Supowit
of the Sethi-Ullman algorithm for **register allocation** Andrew W. Appel Kenneth J. Supowit
www.cs.princeton.edu/~appel/papers/sun.ps

Spill Code Minimization via Interference Region Spilling - Bergner, Dahl.. (1997) (Correct) (7 citations)
Many optimizing compilers perform global **register allocation** using a Chaitin-style graph coloring
optimizing compilers perform global **register allocation** using a Chaitin-style graph coloring
across basic block boundaries, maybe even entire **loops**. By partially spilling live ranges, we can
ftp.inria.fr/INRIA/Projects/a3/lelait/Haiku/SpillPldi97.ps.gz

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (RI) Google (Web) CSB DBLP

79 citations found. Retrieving documents...

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops.* In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283–299, San Francisco, California, June 17–19, 1992. SIGPLAN Notices, 27(7), July 1992.

This paper is cited in the following contexts:

Documents 51 to 79 Previous 50

Advanced Vector Architectures - Espasa (1997) (Correct)

....needed. **On** top of that, program transformations such as loop blocking [PHH89, WL91, KM92, LRW91b, Li95, CM95] have proven very useful to fit the working set of a program into multilevel memory hierarchies. **Introduction 9 Related to data caching, software pipelining [Lam88, GHW90, GAG94, Jai91, RLTS92, Ram94, Rau94] has also contributed to hide memory latency and the penalties associated with cache misses by overlapping several iterations of a single loop.** Decoupling Decoupled scalar processors [SWP86, Smi84, KHC94] have focused on numerical computation and attack the memory latency problem ....

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops.* In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283--299, San Francisco, California, June 17--19, 1992. SIGPLAN Notices, 27(7), July 1992.

Unrolling-Based Optimizations for Modulo Scheduling - Lavery, Hwu (1995) (16 citations) (Correct)

....more simultaneously live values exist than physical registers, spill code must be added and can significantly increase the achieved II of the loop. **In this case, it may be possible to achieve a better final II by increasing the candidate II and attempting to schedule the original loop body again [26].** If a lower bound on the loop s final register requirement for a given II were available, it would be useful during both optimization and scheduling. **During** optimization it could be used to stop optimization before excessive register pressure is generated. **During** scheduling, the candidate II s ....

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker, *"Register allocation for software pipelined loops,"* in Proceedings of the ACM SIGPLAN 92 Conference on Programming Language Design and Implementation, pp. 283--299, June 1992.

Lifetime-Sensitive Modulo Scheduling - Huff (1993) (99 citations) (Correct)

....the successive outputs of an operation can be kept in distinct registers. **In the absence of hardware support, the loop may be unrolled and the duplicate register specifiers renamed appropriately [9] However, this modulo variable expansion technique can result in a large amount of code expansion [18].** A rotating register file can solve this problem without duplicating code. **Consider** saving the series of values generated by an operation in its own infinite pushdown stack. **Old** values can be read out of anywhere in the stack, and new values can be pushed on top, but a value cannot be modified ....

....around a vector of length II. **In** any case, the LiveVector s maximum, MaxLive, is the desired lower bound. **Allocating** registers for a modulo scheduled loop is beyond the scope of this paper. **For an extensive discussion of the problem, including heuristic solutions and empirical results, consult [18].** One of the most remarkable results reported in that paper is the ability of their allocation strategies to almost always achieve the MaxLive lower boundon a schedule s register pressure 4 . Due to that result, this paper approximates a schedule s register pressure with its MaxLive lower bound. ....

B. R. Rau, M. Lee, P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops.* In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283--299, June 1992.

Evaluation of Pseudo Vector Processor based on.. - Nakamura.. (1994) (Correct)

....previous pfld just because its load pipeline has three stages. **On** the contrary, our architecture adopts ordinary waiting mechanism for requested data. **Due** to this fact, our architecture does not need serious changes in the architecture. **Modulo scheduling on rotating register files is proposed in [RLTS92].** In rotating register files, logical register number is apart from physical register number. **In** this point, rotating register files are similar to our slidewindowed registers. **However**, in rotating register files, the total number of physical registers is not increased. **Therefore**, long memory ....

B.R.Rau, M.Lee, P.P.Tirumalai, and M.S.Schlansker, *"Register Allocation for Software Pipelined Loops"*, Proc. ACM SIGPLAN '92 Conf. on Programming Language Design and Implementation, pp283-299, 1992

Register Allocation for Predicated Code - Eichenberger, Davidson (1995) (5 citations) (Correct)

....a framework based on cyclic interval graphs, introducing the notion of time in the register allocator paradigm. **This** additional notion of time is particularly useful for the live ranges of a loop, where live ranges may cross the boundary of an iteration. **Another approach, investigated by Rau et al. [12], proposes a general framework for the allocation of registers in software pipelined loops for various code generation and hardware support schemes.** The second contribution of this paper is a set of heuristics that reduces the register requirements by allowing non interfering virtual registers ....

....For register allocators based on Chaitin s graph coloring framework [9] 10] register allocation for predicated code can be achieved simply by using the refined interference graph instead of the conventional one. **However, several register allocators depart from the graph coloring method [11][12] as graph coloring methods do not provide a notion of time that is particularly useful for the live ranges of a loop, which may cross the boundary of an iteration.** Also, nontraditional constraints such as the one presented in [12] to support various code generation and hardware support schemes are ....

[Article contains additional citation context not shown here]

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. PLDI, pages 283--299, June 1992.

An Integrated Approach to Register Binding and Scheduling - Bart Mesman (Correct)

....to satisfy the timing constraints, software pipelining [2] also called loop pipelining or loop folding, is required. **Previously** [15] we showed that a heuristic like list scheduling for loop pipelining is unable to satisfy the timing and resource constraints even for simple examples. **Rau et al. [11] successfully perform register binding tuned to pipelined loops.** They mention that for better code quality Concurrent scheduling and register allocation is preferable , but for reasons of run time efficiency they solve the problem of scheduling and register binding in separate phases. **Some** ....

B.R. Rau, M. Lee, P.P. Tirumalai and M.S. Schlansker, "*Register allocation for software pipelined loops*", Proc. of the SIGPLAN '92 conf. on Programming language design and implementation, pp. 283-299, June 1992

Minimizing Register Requirements under.. - Govindarajan, Altman.. (1994) (40 citations) (Correct)

....Huff s Slack Scheduling [9] Wang, Eisenbeis, Jourdan and Su s FRLC [23] and Gasperoni and Schwiegelshohn s modified list scheduling [6] Experimental results show that the method described in this paper performed significantly better than these methods. **1 Introduction Software pipelining [1, 4, 9, 11, 12, 13, 17, 18, 22] has been proposed as an efficient method for loop schedul This work was supported by research grants from NSERC (Canada) and MICRONET Network Centers of Excellence (Canada) To Appear in the Proceedings of the 27th Annual International Symposium on Microarchitectures (MICRO 27) San Jose, ...**

....in Section 7. 2 Exploiting the Space of Software Pipelined Schedules 2. **1 An Example We introduce the notion of rate optimal schedules under resource constraints, and illustrate how to search among them the ones which optimize the register usage with the help of a simple example loop taken from [18].** The loop L (in the C language) is: for (i = 0; i n; i) f s = s a[i] a[i] s s a[i] g The dependence graph for the loop L is depicted in Figure 1. **S0** S1 S2 S3 S4 S5 Figure 1: Dependence Graph of Loop L Consider an architecture with 3 pipelined homogeneous function units. **Assume** ....

[Article contains additional citation context not shown here]

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops*. In Proc. of the SIGPLAN '92 Conf. on Programming Language Design and Implementation, pages 283--299, San Francisco, CA, Jun. 17--19, 1992.

Towards Identifying and Monitoring Optimization Impacts - Way, Pollock (1997) (2 citations) (Correct)

....is rarely gathered and exploited in the optimizer s strategy. **There are isolated instances where this information is used to good effect, such as when combining instruction scheduling and register allocation [3, 5, 6, 19, 20, 30, 33, 34] or software pipelining and register allocation [16, 17, 21, 23, 27, 32, 38, 44].** While these techniques can improve program performance, they focus narrowly on the interaction of a single pair of optimizations, rather than more generally on the entire collection of optimizations to be applied to a program. **Provided** that enough useful information can be gathered and analyzed ....

....of balance among the levels of demand for specific machine resources of particular interest to the two phases, and the supply and configuration of the target machine s resources. **The most well known examples of this work focus on the interactions between software pipelining register allocation [16, 17, 21, 23, 27, 32, 38, 44], instruction scheduling and register**

allocation [3, 5, 6, 19, 20, 30, 33, 34] instruction scheduling and cache usage [28] and scalar replacement and register allocation [8] All have in common the goal of creating a good match between the program characteristics, such as instruction placement ....

B. R. Rau, M. Lee, P. P. Tirumalai, and M. S. Schlansker. *Register allocation for software pipelined loops.* In ACM SIGPLAN Conference on Programming Language Design and Implementation, 1992.

Non-Consistent Dual Register Files to Reduce Register Pressure - Llosa, Valero, Ayguade (1995) (8 citations) (Correct)

....Chaitin s technique based on graph coloring[14] Register allocation for software pipelined loops presents additional problems leading to unconventional solutions. **How to allocate registers for modulo scheduled loops is beyond the scope of this paper (for an extensive discussion of the problem see [15]) The Wands Only strategy combined with the First Fit allocation schema have been chosen to allocate registers.** Wands Only is the strategy that has the lowest empirical complexity, and the one that obtains the more optimal results in terms of number of registers. **For** this strategy all the ....

....M3 and A4. **The** results of M3 are used by operation A4; since A4 has been scheduled in 5 We have chosen this example because it is very simple to calculate the registers required by the schedule. **For an extensive discussion of the register allocation problem for software pipelined loops see [15].** VALUE L1 L2 M3 A4 M5 A6 Allocation GL LO LO RO RO RO Lifetime 13 7 6 6 6 4 Table 3: Allocation requirements of values for example loop. **the** left cluster, the values produced by M3 could be allocated as left only values. **The** results of A4 are used by operation M5; since M5 has been scheduled ....

B.R. Rau, M. Lee, P. Tirumalai, and P. Schlansker. *Register allocation for software pipelined loops.* In Proceedings of the ACM SIGPLAN '92 Conference on Programming Language Design and Implementation, pages 283--299, June 1992.

Constraint Driven Approach To Loop Pipelining And.. - Mesman, Strik.. (1998) (Correct)

....to satisfy the timing constraints, software pipelining [2] also called loop pipelining or loop folding, is required. **Previously** [15] we showed that a heuristic like list scheduling for loop pipelining is unable to satisfy the timing and resource constraints even for simple examples. **Rau et al. [11] successfully perform register binding tuned to pipelined loops.** They mention that for better code quality Concurrent scheduling and register allocation is preferable , but for reasons of run time efficiency they solve the problem of scheduling and register binding in separate phases. **Some** ....

B.R. Rau, M. Lee, P.P. Tirumalai and M.S. Schlansker, "*Register allocation for software pipelined loops*", Proc. of the SIGPLAN '92 conf. on Programming language design and implementation, pp. 283-299, June 1992

A Scalar Architecture for Pseudo Vector Processing based on.. - Nakamura Hiroshi (1 citation) (Correct)

....three stages. **Compared** with i860 architecture, our architecture includes ordinary waiting mechanism for requested data and successfully closes the growing gap between processor and memory speed without serious changes in the architecture. **Modulo scheduling on rotating register files is proposed in [RLTS92].** In rotating register files, logical register number is apart from physical register number. **This** is similar to our slide windowed registers. **However**, in rotating register files, the total number of physical registers is not increased. **Therefore**, long memory access latency cannot be hidden. **This** ....

B.R.Rau, M.Lee, P.P.Tirumalai, and M.S.Schlansker, "*Register Allocation for Software Pipelined Loops*", Proc. ACM SIGPLAN '92 Conf. on Programming Language Design and Implementation, pp283-299, 1992

Array Data Flow Analysis for Load-Store Optimizations in.. - Bodík, Gupta (1995) (2 citations) (Correct)

No context found.

B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker, "*Register Allocation for Software Pipelined Loops*," Proc. of the SIGPLAN Conference on Programming Language Design and Implementation, San Francisco, California, pages 212-223, June 1992.

Documents 51 to 79 Previous 50

Online articles have much greater impact More about CiteSeer Add search form to your site Submit documents Feedback